architecture. In a conventional processor having a data hit/miss predictor, when a data cache miss is predicted, no instructions (in case of an in-order dispatch engine), or only those that do not depend on the missing data (in case of an out-of-order dispatch engine) can execute. In any case, the processor resources might be idle for several cycles until the missing data is available. In multistreaming processors those idle cycles can be used to execute other instructions from other threads since they do not depend on the missing data. Thus, for a multistreaming processor, the benefits of a data cache hit/miss predictor are twofold as shown in Figure ~~3~~ 4.--

In the claims:

All of the claims standing for examination are presented below. Claims 1, 4, 6, 9 and 12 are herein amended in the present response.

1. (Currently Amended) In a multi-streaming processor having a data cache, a system for fetching instructions from individual ones of the multiple streams to a pipeline, comprising:

a fetch algorithm for selecting from which stream to fetch instructions; and

a hit/miss predictor for forecasting whether instructions will hit or miss the data cache;

wherein the prediction by the ~~hit-miss~~ hit/miss predictor is used by the fetch algorithm in determining from which stream to fetch.

2. (Original) The system of claim 1 wherein a hit prediction precipitates no change in the fetching process.

3. (Original) The system of claim 1 wherein a miss prediction results in switching fetching to a different stream.

4. (Currently Amended) The system of claim 1 wherein the ~~hit-miss~~ hit/miss predictor determines a hit probability, and the probability is used by the fetch algorithm in determining from where to fetch next instructions.

5. (Original) The system of claim 1 wherein the forecast of the hit/miss predictor is also used by a dispatch algorithm in selecting instructions from the pipeline to dispatch to functional units.

6. (Currently Amended) A multi-streaming processor comprising:

a data cache;

a fetch algorithm for selecting from which stream to fetch instructions; and

a hit/miss predictor for predicting whether instructions will hit or miss the cache;

wherein a prediction by the ~~hit-miss~~ hit/miss predictor is used by the fetch algorithm in determining from where stream to fetch.

7. (Original) The processor of claim 6 wherein a hit prediction precipitates no change in the fetching process.

8. (Original) The processor of claim 6 wherein a miss prediction results in switching fetching to a different stream.

9. (Currently Amended) The processor of claim 6 wherein the ~~hit-miss~~ hit/miss predictor determines a hit probability, and the probability is used by the fetch algorithm in determining from where to fetch next instructions.

10. (Original) The processor of claim 6 wherein the forecast of the hit/miss predictor is also used by a dispatch algorithm in selecting instructions from the pipeline to dispatch to functional units.

11. (Original) In a multi-streaming processor having a data cache, a method for fetching instructions from individual ones of multiple streams as instruction sources to a pipeline, comprising the steps of:

(a) making a hit/miss prediction by a predictor as to whether instructions previously fetched will hit or miss the data cache; and

(b) if the prediction is a miss, altering the source of the fetch.

12. (Currently Amended) The method of claim 11 wherein the ~~hit-miss~~ hit/miss predictor determines a hit probability, and the probability is used in determining fetch source.

13. (Original) The method of claim 11 wherein the forecast of the hit/miss predictor is also used by a dispatch algorithm in selecting instructions to dispatch to functional units.